Minimizing Churn Rate through Analysis of Financial Habits

Vineet Unnithan

Contents

1. Introduction:

Subscription products are often the main source of revenue for companies in many industries. These products can be a 'single-level' subscription or 'multi-level' memberships. Regardless of what level of products or which industry, companies almost always try to minimize customer churn (subscription cancellations). Hence, in order to retain their customers, these companies need to identify behavioral patterns that act as catalyst in disengagement with the product.

Market: The target audience is the entirety of company's subscription base. They are the ones that the company aims at retaining.

Product: The subscription products that the customers have already enrolled in, can provide value that users cannot have imagined, or that they may have forgotten.

Goal: The objective of the model is to predict which users are likely to churn, so that the company can focus on re-engaging these users with the product. These efforts can be sending email reminders about the benefits of the product, especially focusing on features that are new or that the user has shown to value.

1.1 Business Challenge:

In this case study, we are working for a fintech company that provides a subscription product to its users, which allows them to manage their bank accounts, provides them with personalized savings, coupons, informs them about latest low-interest rates loans available for them and educates them with best available methods to save money.

We are in charge of identifying users who are likely to cancel their subscription so that we can start building new features they might be interested in. These features can increase engagement and interest of users towards the product.

1.2 Data:

By subscribing to our membership, our customers have provided us with data on their finances, as well as how they handle their finances through our product. We also have some demographic information which we acquired from them during the sign-up process.

Financial data can often be unreliable and delayed. As a result, the companies can sometimes build their marketing model using only the demographic data and data related to finances handled through the product itself. Therefore we will be restricting ourselves to only using that type of data. Furthermore, product related data is more indicative of what new features we should be creating as a company.

2. Exploratory Data Analysis:

As a part of any model development, it is necessary that we first examine the data and perform exploratory data analysis so that our dataset helps the model to predict accurately and perform optimally.

As seen from the dataset, it has 27000 observations and 31 columns.

```
In [3]: dataset.head()
```

Out[3]:

	user	churn	age	housing	credit_score	deposits	withdrawal	purchases_partners	purchases	cc_taken	...	waiting_4_loan	cancelled_loan	received_loan
0	55409	0	37.0	na	NaN	0	0	0	0	0	...	0	0	0
1	23547	0	28.0	R	486.0	0	0	1	0	0	...	0	0	0
2	58313	0	35.0	R	561.0	47	2	86	47	0	...	0	0	0
3	8095	0	26.0	R	567.0	26	3	38	25	0	...	0	0	0
4	61353	1	27.0	na	NaN	0	0	2	0	0	...	0	0	0

5 rows × 31 columns

```
In [4]: dataset.shape
```

Out[4]: (27000, 31)

Also let us see the descriptive statistics of the different features in the datasets. This gives us an idea of how well distributed our dataset is.

```
In [5]: dataset.describe()
```

Out[5]:

	user	churn	age	credit_score	deposits	withdrawal	purchases_partners	purchases	cc_taken	cc_recommended
count	27000.000000	27000.000000	26996.000000	18969.000000	27000.000000	27000.000000	27000.000000	27000.000000	27000.000000	27000.000000
mean	35422.702519	0.413852	32.219921	542.944225	3.341556	0.307000	28.062519	3.273481	0.073778	92.625778
std	20321.006678	0.492532	9.964838	61.059315	9.131406	1.055416	42.219686	8.953077	0.437299	88.869343
min	1.000000	0.000000	17.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	17810.500000	0.000000	25.000000	507.000000	0.000000	0.000000	0.000000	0.000000	0.000000	10.000000
50%	35749.000000	0.000000	30.000000	542.000000	0.000000	0.000000	9.000000	0.000000	0.000000	65.000000
75%	53244.250000	1.000000	37.000000	578.000000	1.000000	0.000000	43.000000	1.000000	0.000000	164.000000
max	69658.000000	1.000000	91.000000	838.000000	65.000000	29.000000	1067.000000	63.000000	29.000000	522.000000

8 rows × 28 columns

2.1 Missing Value Analysis:

We need to check for missing values before we plot any visualizations. As we checked, there were missing values found in the features 'age', ' credit_score', 'rewards_earned'. Their counts are given below:

```
In [7]: dataset.isna().sum()

Out[7]: user                        0
        churn                       0
        age                         4
        housing                     0
        credit_score             8031
        deposits                    0
        withdrawal                  0
        purchases_partners          0
        purchases                   0
        cc_taken                    0
        cc_recommended              0
        cc_disliked                 0
        cc_liked                    0
        cc_application_begin        0
        app_downloaded              0
        web_user                    0
        app_web_user                0
        ios_user                    0
        android_user                0
        registered_phones           0
        payment_type                0
        waiting_4_loan              0
        cancelled_loan              0
        received_loan               0
        rejected_loan               0
        zodiac_sign                 0
        left_for_two_month_plus     0
        left_for_one_month          0
        rewards_earned           3227
        reward_rate                 0
        is_referred                 0
        dtype: int64
```

Since the number of missing values in the 'age' feature are only 4, it makes perfect sense to remove those observations completely as this is not going to affect our dataset in any way.

Coming to others like 'credit_score' and 'rewards_earned', these features have too many missing values, 8031 and 3227 respectively. Hence, it is best for our model that we exclude these features completely as they would not be playing an important part in explaining the variance in the dataset.

After a complete removal of all missing values, our dataset is clean from NA values.

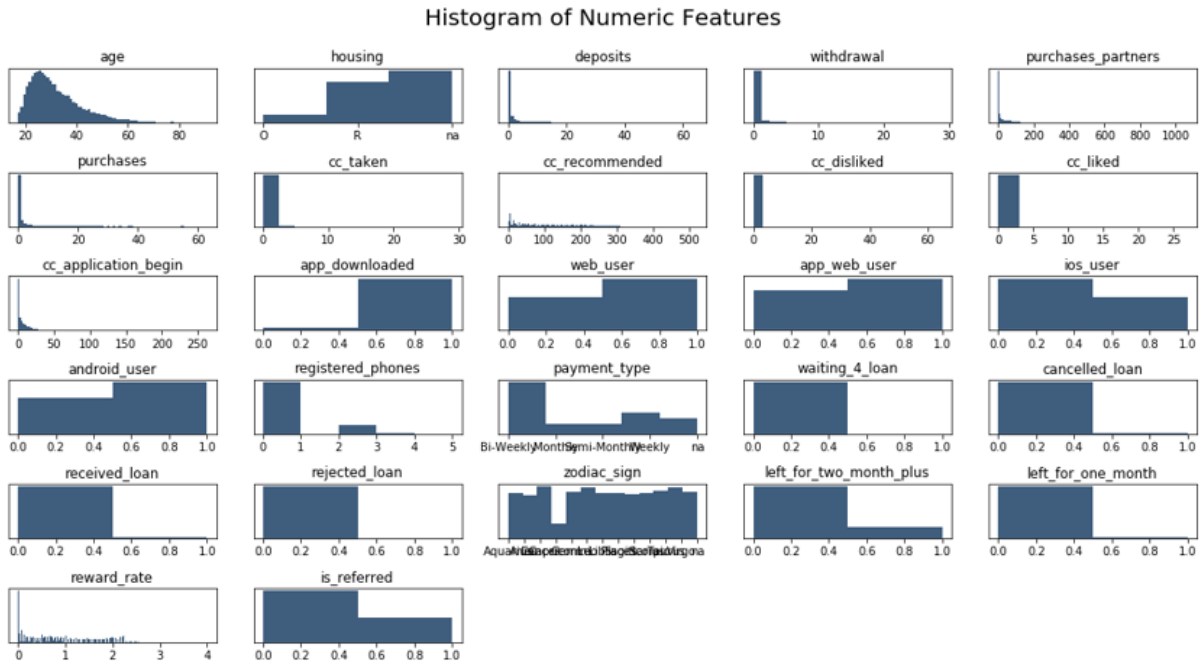
In [11]:
```
dataset.isnull().sum()
```

Out[11]:
```
user                       0
churn                      0
age                        0
housing                    0
deposits                   0
withdrawal                 0
purchases_partners         0
purchases                  0
cc_taken                   0
cc_recommended             0
cc_disliked                0
cc_liked                   0
cc_application_begin       0
app_downloaded             0
web_user                   0
app_web_user               0
ios_user                   0
android_user               0
registered_phones          0
payment_type               0
waiting_4_loan             0
cancelled_loan             0
received_loan              0
rejected_loan              0
zodiac_sign                0
left_for_two_month_plus    0
left_for_one_month         0
reward_rate                0
is_referred                0
dtype: int64
```

2.2 Features Histograms:

As a part of our EDA, it is necessary to plot the histograms for the numeric variable to understand their frequency distribution. Also it helps us to understand which features are highly skewed.

We have excluded ‘user’ and ‘churn’ (which is our response variable) from histogram plotting since values for ‘user’ variable are random and hence does not make sense to plot its distribution.



As seen from the above histograms, we can derive a few inferences from the distribution.

As seen from the ‘age’ histogram, we can see that it is particularly right skewed and starts from 18, since it is practically the age at which one can get eligible for the subscription product. We can derive that most of our customers are between the age groups 20-40.

From the ‘housing’ histogram, we can see that majority of the population are not owners, but have rented housing and a very few have their own house. Also a major portion of them are unclassified.

The next three ‘deposits’, ‘withdrawal’ and ‘purchases’ are right-tailed and majority of them have zero value which means zero deposits, withdrawals or purchases have been made. This usually happens in most of the cases, in the first couple of months of enrollment. Activity usually stays low in the first couple of months till the customers start trusting the system and then start making deposits, withdrawals or purchases. Also it is important to note that the

engagement of the product is not shown by these features but actually on how they use the product.

The next three features ‘cc_taken’, ‘cc_disliked”, ‘cc_liked’ have taken binary values and hence all the distribution is concentrated between 0 and 1.

However, ‘cc_recommended’ has somewhat an evenly normally distributed plot.

We can see that most of our customers have either the app downloaded or are a web user or use both. This is explained by the ‘app_downloaded’, ‘web_user’ and ‘app_web_user’ plots respectively.

To find the distribution % for all features having binary values, we will be using ‘pie charts’ to understand it more clearly.

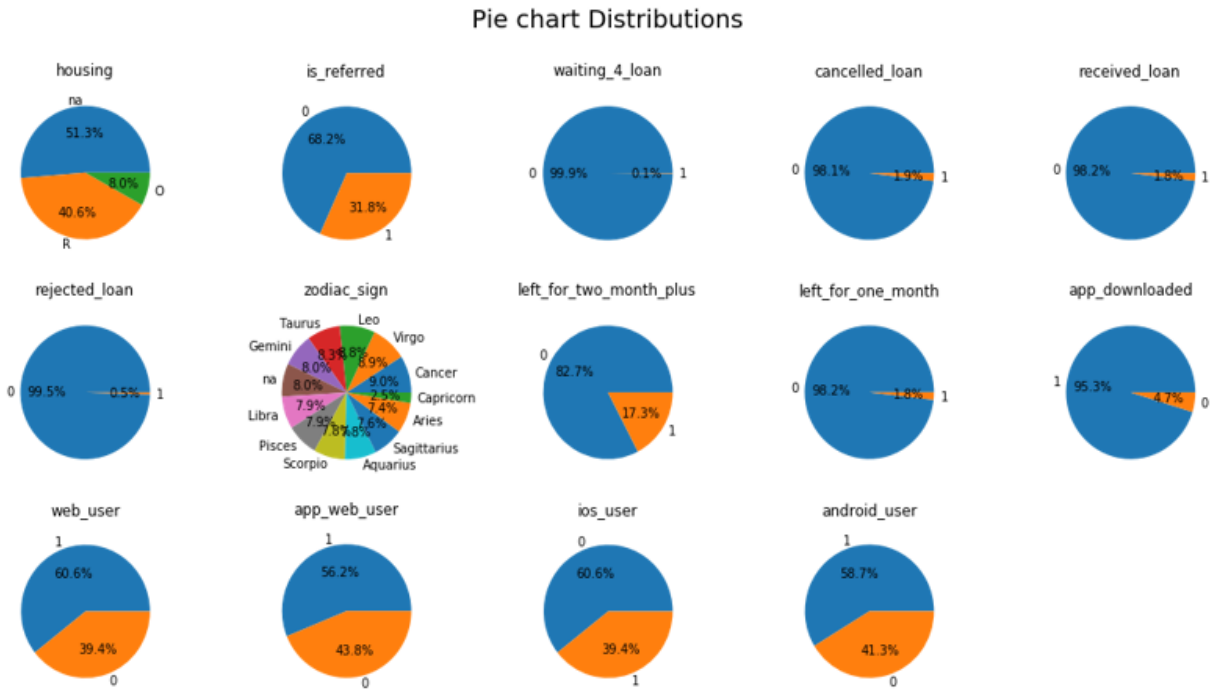
Next, we can derive that majority of the people have one phone registered with us and only a few have more than one.

From ‘payment_type’ plot, we can identify that majority of our customers are paid ‘bi-weekly’ and ‘weekly’ mostly and few are paid monthly and semi-monthly.

From ‘zodiac_sign’, we can make an interesting observation which is that the whole plot is almost evenly distributed except in case of ‘Capricorn’ which is not highly represented.

2.3 Pie Chart Distribution:

Since we have a lot of features with binary values, we will plot pie charts to understand if their distribution is uneven, if so then how uneven are they and to check whether or not those uneven subsets have or contain both values of the response variable. This is essential to avoid unusual bias in the model.



As seen above, the ‘housing’ pie plot can be broadly classified into two categories namely undisclosed and disclosed (includes rented and owned). We have a nice distribution and nothing suspicious about it.

As is the same with others like ‘is_referred’, 'app_downloaded', 'web_user', 'app_web_user', 'ios_user', 'android_user’. Their distribution seems nice and unsuspicious of any bias.

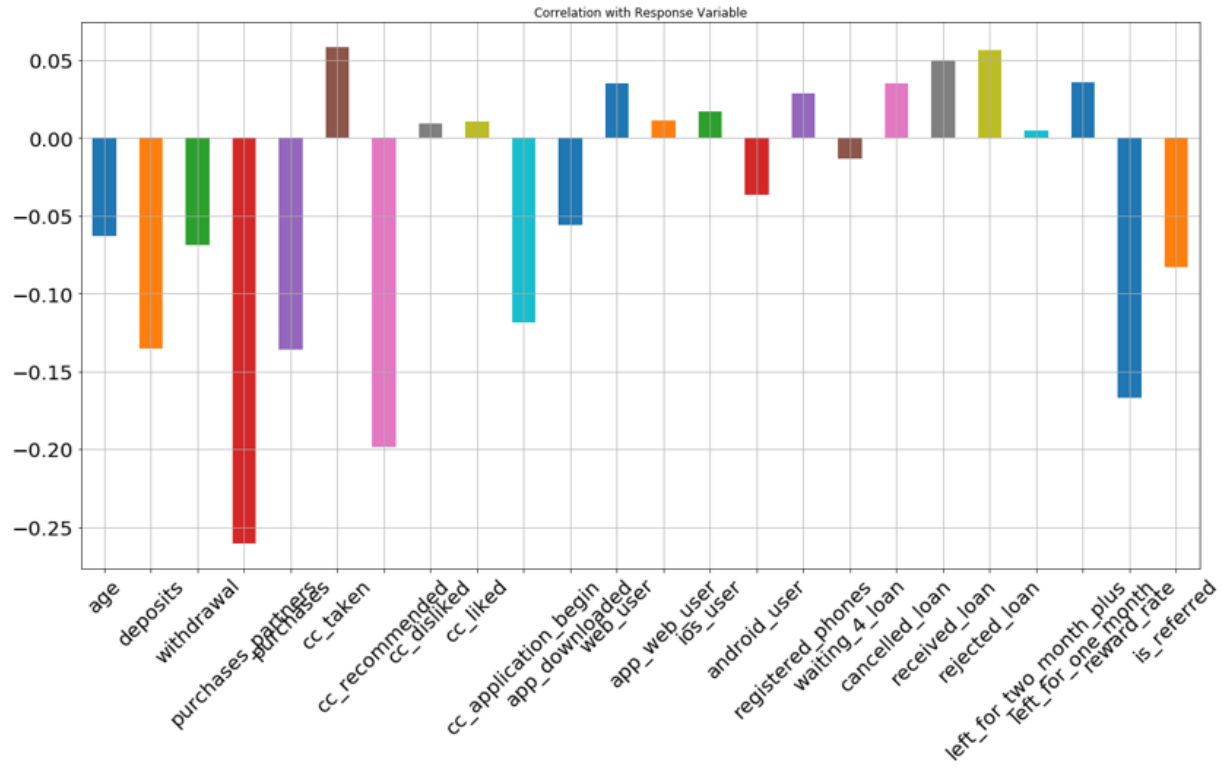
However, we have some features for e.g ‘waiting for loan’ where the small subset is too small and we need to verify if this feature is useful for us to be included in the model and check if this subset has both responses of our target variable ‘churn’ included in it. For this, we will subset them from the dataset and then do a value count distribution for the response ‘churn’ variable.

```
In [26]: dataset[pie_data.waiting_4_loan == 1].churn.value_counts()

Out[26]: 0    27
         1     8
         Name: churn, dtype: int64
```

As we can see there is little or no bias here. Similarly for all other features, we found that the distribution were not very uneven and had both values of the response variable ‘churn’.

2.4 Correlation Plot:

We need to analyze the correlation of the independent features with the response variable to understand how they affect the churn behavior of the customers.



As seen from above, ‘age’ is negatively correlated with our response variable ‘churn’. This means the younger the age, it is more likely for a customer to churn. The same is applicable for the next 3 features ‘deposits’, ‘withdrawal’ which indicates the less activity a customer has, the more likelier they are to churn. The same can be said aabout ‘purchases_partners’ and ‘purchases’.

The next ‘cc_taken’ shows a different picture altogether. It shows that if a customer has taken a credit card with us the more likely they are to churn. This actually gives us insight

that customers aren't happy with the features we are offering them and maybe we need to work on it to prevent them from churning.

The next feature'cc_recommended' clearly indicates that the lesser the number of times cc is recommended, more likely it is that the customer is going to churn. This makes perfect sense in logic.

The next 2 features 'cc_liked' and cc_disliked' have low correlation values and are not very strong. Hence we can overlook them for now.

The feature 'cc_application_begin' tells us that the lesser applications we get, more likely for customers to churn which also makes perfect sense. We need to increase cc_applications so that we retain our existing customers.

Coming to 'app_downloaded' it also tells us that the more customers have downloaded the app, the less likely they are to churn.

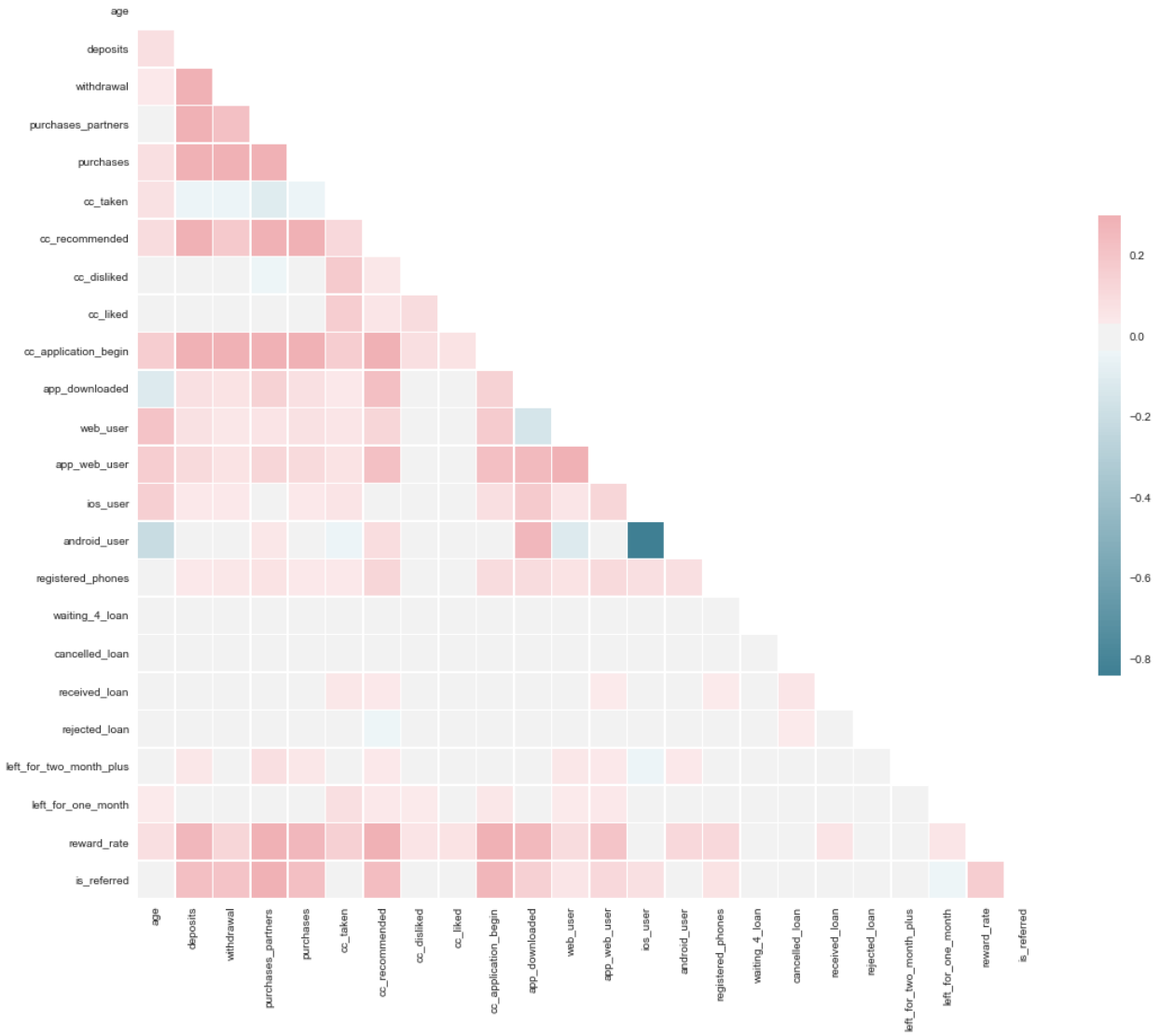
As we move on, we can observe that the strength of the bars is relatively low (<0.5) and hence they are not displaying strong correlation with our response variable.

The next features 'waiting_for _loan', 'cancelled loan', 'rejected loan' and 'rejected loan' are all positively correlated which indicated that if a customer performs this action, more likelier they are to churn.

The last feature' is_referred' clearly shows that the likelihood of churn decreases if they are referred by somebody.

2.5 Correlation Matrix:

We need to plot the correlation matrix to understand and analyze the correlation between the independent variables.



We can see from the legend above that the strongest positive correlation is of 0.2 which is not the strong; however on the other side, the strongest negative correlation is -0.8 which is very strong and almost close to 1.

This is between android_user and ios_user which in fact, makes a lot of sense as, if a person is an android user it is most likely they may not be an ios_user. This correlation is not exactly 1 to 1 as there may be instances where a person has both android and ios phones. But still, these features display a very strong correlation and hence it makes sense to include only one of them in our model building.

Apart from this, there is a correlation which is not clearly visible but we can infer that simply by looking at the features. We see that the feature 'app_web_user' can be true only if 'app_user' and 'web_user' features are true. Hence we can say that 'app_web_user' is a function of the two features and not an independent feature. Thus, since we want only independent variables to explain our response variable, we will remove this feature from the dataset before building the model.

This is a good point to note that the correlation matrix cannot show all the correlations among the independent variables and that, we have to observe the dataset and make our own conclusions out of it.

2.6 Data Preparation:

Before we start building our model, we need to do some further pre-processing.

We will first remove the 'user' column as we do not need it during our model building phase, but we would need it later to identify the customers.

We will be using the one-hot encoding method so that we can convert the categorical variable into their own independent binary columns as python cannot process strings as independent variables. Once done, we also need to be careful to avoid the dummy variable type. This is caused due to correlated features created after one-hot encoding of the categorical variables. So in order to avoid this trap, we are going to remove one of the hot encoded columns from each categorical variable.

Once this is done, we can then move onto splitting the dataset into training and test, simply because we want our model to be efficient and not overfitting. We will apply the model on the training set and then compare the predicted values with the test set to check the performance of our model.

2.7 Balancing and Feature Scaling:

After we have split the dataset, we need to perform balancing on the training and test set of the response variable to ensure that we have equal number of values of the response variable in both sets and hence no bias.

Once balancing is done, we also need to perform feature scaling (normalization) to ensure all numerical variables are in the same range.

3.0 Model Building:

We have successfully cleaned, balanced and feature scaled the dataset. Now we will build our logistic regression model on the training dataset since this is a classification based problem.

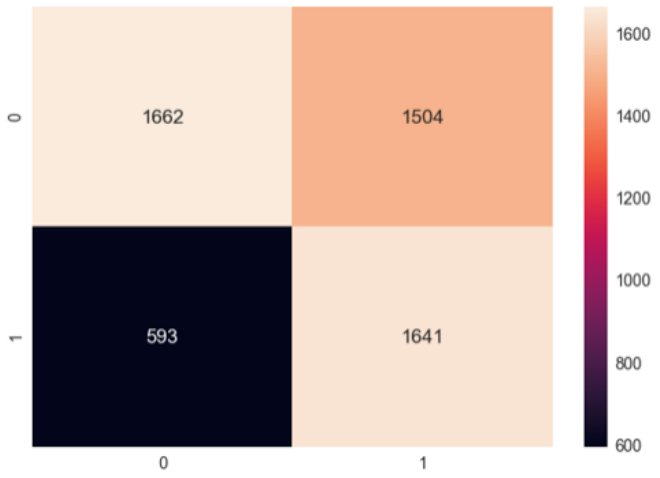
Once our model was built, we calculated how well our model is performing based on certain metrics like accuracy, precision, f1 score and recall.

The values for each of the scores are given below:

```
In [68]: cm = confusion_matrix(y_test, y_pred)
         accuracy_score(y_test, y_pred)
Out[68]: 0.6116666666666667

In [69]: precision_score(y_test, y_pred)
Out[69]: 0.5217806041335453

In [70]: recall_score(y_test, y_pred)
Out[70]: 0.73455684870188

In [71]: f1_score(y_test, y_pred)
Out[71]: 0.6101505856107082
```

Apart from that, we have also plotted the confusion matrix to get a better idea where our model has misclassified the most errors.

```
In [72]: #Plotting the confusion matrix

         df_cm = pd.DataFrame(cm, index = (0,1), columns = (0,1))
         plt.figure(figsize=(10,7))
         sns.set(font_scale= 1.4)
         sns.heatmap(df_cm, annot=True, fmt='g')
Out[72]: <matplotlib.axes._subplots.AxesSubplot at 0x20bf02aa240>
```



As seen from the above confusion matrix, our model is better at predicting the negative values; however does not perform that well in case of the negative values.

We also applied K-fold cross validation to check if the accuracies are constant across all our features and we found that they were almost equal across all of the columns.

We also analyzed the coefficients derived from the classification model to understand which features are most important in prediction of the response variable.

We found that most of them had a low value of coefficients except ‘purchases’, ‘purchases_partners’, ‘deposits’, which had high values. This means that these features explained our response variable with utmost significance.

3.1 Feature Selection:

We want to find the features that explain the most significance for the response variable. We have a total of 40 columns and we need to narrow it down and find the most important ones.

Hence we decided to select the top 20 columns that play an important role to explain our target variable.

However after selection of these 20 top feature and running the model once again, we did not find any major significant difference in metrics scores as all of them hardly changed by a significant amount.

Hence we came to a conclusion that our aim was to build a model that would predict if a customer was going to churn. However, most of the features that we have, have nothing to do with the ability to churn and are pretty much useless in explaining this. However it was important to validate this belief and hence we selected the top 20 features and built a model using them as these were the features on which the most predictive power was lying.

3.2 Model Conclusion:

To finalize our model, we first decided to gauge the coefficients of the 20 features that we selected to build the model on.

Also we got our final results which contains the user identifier, the actual churn value and the predicted churn value.

4.0 Conclusion:

Our model has given us an indication of which users are likely to churn. Unlike the first model we built, we have no timeframe on what we can allow the response variable to take place. In other words, we are not predicting if the customer is going to churn in one day, one month or one year. We have left this open-ended because we are focused on only gauging the features that indicate disengagement with the product, and not the exact manner (like timeframe) in which users will disengage. The reason we did not put a timeframe in this project was because we wanted to explore the features may indicate if the user is going to churn or not, no matter when. We wanted to identify those fields whose impact on churn happened after a long time of enrollment. If we would have put a time-frame limit on the response variable, then the features that cause a customer to churn after the timeframe would not have been identified since the users who churn post the timeframe would have been classified as not churned.

Regardless of this choice, we can still leverage the model as usual because we can gauge who is even a bit likely to churn from the product and as a result we can build new features for the users who are losing interest. This is the main benefit of this model. If after creating new product features, we start seeing our model predict less users are going to churn, then it is safe to assume that our customers are feeling more engaged with what we are offering them.

We can further fine tune this approach by doing research on the opinion of our users of our products, for e.g. through polls. If, instead, we want to fine tune the model to predict churn more accurately, above the 62% accuracy we obtained, then we can add a time dimension to the response variable churn. This model will predict strictly who is leaving us in a time frame rather than those who show certain symptoms of losing interest in our product.